# Inspecting Monocular Depth Perception

Bahaa Aldeeb

***Abstract*—Sensing distance from surroundings is essential for a mobile robot to be able to navigate around environments safely. This paper focuses on the autonomous wheelchair platform Vulcan. Vulcan hosts two 2D LIDAR sensors and a camera, which it relies on to navigate environments riddled with staircases, glass walls, and overhanging objects like tables and chairs. The LIDARS used are not directed towards the ground so they cannot identify drops, and they cannot see table tops or skinny reflective table legs, which are common in the campus environment the chair operates in. This requires the camera to be able to identify depth variations and label those obstacles and drops as hazards.**

**In this paper different methods used for depth perception for monocular vision are explored. CNNs are touched on then the ideas that allow those CNNs to perceive depth are explored. A method that relies on optical flow is implemented along with a probability grid that helps solidify the hypotheses assumed by that method. Results of that method are highlighted along with some of its weaknesses. Potentially usable methods are also listed and discussions on how those methods could append the former concludes the paper.**

## I. Introduction

Many cues are leveraged by today's algorithms to perceive depth from single or multiple views. Cues that rely on intensity gradients, line intersections, edges, and shading benefit from prior knowledge which is what makes CNNs very effective at learning such features. For those reasons, the paper initially focused on CNNs, but deviated after realizing the technical limitations of the platform and the difficulty of finding sufficient training data. When tested in some areas of the campus with insufficient lighting, unconventional objects, and glass walls the network struggled. Methods exploited by the network and their limitations are discussed in methodology.

The paper then focuses on algorithms that focus on optical flow cues. Bayesian inference is explored as a way to merge multiple predictions and make the results of these methods more reliable. The paper describes how optical flow and camera motion could be used for depth perception. Results of using cues from sudden change in optical flow are described then in the results section. Suggestions on how to use the localization capability of the chair to append optical flow is then stated in the discussion section.

## II. Methodology

## III. Using Convolutions Neural Networks

CNN usage for vission applications in general and monocular depth perception particularly is now more common because of the increasing ability of Convolutions Neural Networks to learn complex functions, which allows for relating various image features without having to program those features and their relations explicitly. Such relations might be of edges and corners or even certain shapes of known objects and their learned relative sizes. tailoring network structures for specific applications further helps them learn what is relevant for that application. Training stays a common neural network hurdle, but some research presented workarounds for collecting ground truth data using conventional distance sensors for training. Below is a brief discussion on how some monocular depth networks deal with the difficulty of training such networks, and a deeper dive into a network that leverages binocular vision, and the idea behind using two images to learn monocular depth perception.

### A. *Architectures Assisting Depth Estimation*

ResNets and VGG networks improved the role of CNNs in vision applications. Some network architectures tailor specifically for the application of depth perception. In [1] similar to Eigen et al the network architecture was tailored to allow the network to learn course features and Gradients first then combining them as layers along with the intensity image to produce a dense depth image. This techniques assured that the network leverages the information provided by edges for depth prediction. The architecture also allowed for utilize courser wider features which depth might be easier to learn compared to dense images.

Data availability proves to be a serious application specific issue, especially for depth networks. Training some depth networks requires collecting ground truth depth data that is commonly obtained using LIDAR sensors or Kinect sensors. Although those 3D space imaging technologies have improved, they do not handle reflective or transparent surfaces properly. Data collected by the two sensors is also limited in distance and lighting and could be less dense. To avoid issues with collecting such data [2] suggests leverages synthesized data, which yields good results but has its limitations.

Monocular depth perception is important and clearly depended on by animals along with optical flow and binocular vision cues, but binocular perception proves to be much more reliable for calculating ground truth depth when compared to other monocular cues. The next section dives deeper into how one network leverages that to learn monocular depth.

### *B. CNNs & Epipolar Constraints*

To eliminate the use of ground truth depth data for training, Godard et al presented a self-supervised CNN that produces a dense depth image as a by-product of learning how to match the different sides of binocular images [3]. The network is trained using binocular vision to learn how to predict disparity. Disparity is then used to reconstruct the the right image from the left, and vice versa, by exploiting epipolar constraints. Loss is calculated based on the difference of the reconstructed image and the original. Transforming disparity predictions into a depth predictions is then trivial.

Epipolar constraints are intrinsic geometric properties between two views that are independent of the external geometry [4]. Those constraints associate the projections of a single features onto different views. When the depth of a feature varies along the line formed by it's intersection with the focal point of one image's perspective, the position of that feature moves along a line in another image plane. This line presents the constrained path possible as the feature changes depth. That line of motion of the specific feature is called the epipolar line. This suggests that using a calibrated rig-a rig with known camera positions and properties-allows for the calculation of depth of a feature using it's disparity between the two images. That also ensures that one image can be predicted using depth and the relative positions of the two cameras. The following equation expresses the relation, and is further explained in [4]:

$$x' = K'RK^{-1}x + \frac{K't}{z}$$

Where $x$ and $x'$ are the 2D homogeneous coordinates of a feature in the first and second image planes, $K$ and $K'$ are the calibration matrices of the two cameras, $R$ and $t$ are the rotation and translation that transforms pixels from the first to the second perspective. In binocular vision $R$ is ideally an identity matrix and $t$ simply reflects the baseline or distance between the two focal points.

Predicting the disparity of a pixel between the two perspectives allows for predicting one image from another by applying that disparity to each pixel. That is represented generally in [3] as:

$$x' = x(d^l)$$

Comparing the predicted pixel position using equation above and its real position in the other view produces a loss function capable of being used to train the network to predict $d^l$. After the network predicts disparity, depth can be calculated using the cameras' relative positions:

$$z = b\frac{f}{d}$$

Where $b$ is the baseline, $f$ is the focal length, $d$ is the disparity of the pixel, and $z$ is the depth.

### *C. Results In BBB Building*

Although those methods allows for high accuracy and extensive training with relatively little effort to collect the data it still has the downsides associated with using a neural network. Effectiveness of this network relies on the data used for training it. Convolutional networks rely on the assumption that the data provided at test time is similar to that used in training, and the networks above are no exception. The results acquired from using models of the latter network trained on the Kitti main split and Eigen split of the Kitti database were discouraging. The depth image did not show clear edges between different relevant object as shown in the figure 1

Figure 2 further highlights the weaknesses of the models and the requirement of binocular vision and retraining for them to be useful. This project diverged from neural networks because of time and resource constraints. Binocular training is a worthy realm to explore if it is possible to fit the robot with binocular vision at training time.

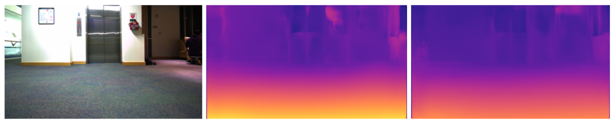

**Figure 1:** The figure shows the original image, depth from Kitti trained model, and depth from Eigen trained model from left to right. Although one can notice the elevator it is not prominent. Depth is represented by colors with red showing the closes object gradually increasing in red through yellow then blue and black.

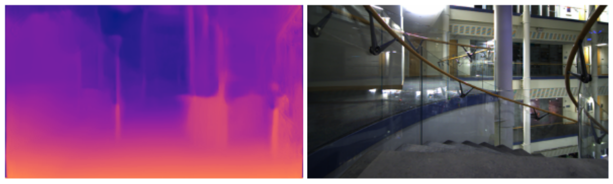

**Figure 2:** Round staircase in the BBB Building at the University of Michigan. Depth predicted for that image proved to be much less informative and clear. The color spectrum shows increasing depth starting with red representing the least depth to black representing the farthest.

## IV. Single View Geometry

Plenty of cues give away depth from monocular vision. Methods that use prior knowledge of object sizes, assumptions about angles and line directions, object height in image, shading, object obscurity, and others rely on learned relations to hypothesize on the depth of certain features. Without machine learning it is hard to program such inferences robustly, non the less get a dense depth image out of those ques. Disparity and motion cues are more common in applications like Vulcan.

## V. Image Sequence

Optical flow and disparity, coupled with knowledge about the displacement of the camera provide enough data to calculate depth. Sudden changes in optical flow alone could also provides useful cues about sudden depth variation but such calculations require image sequences.

### A. *Optical Flow*

Optical flow is the apparent displacement of objects in an image. The term was initially introduced by the psychologist James J. Gibson and further studied by his followers who noticed its significance in helping animals perceive their motion and the motion and shapes of objects around them [5]. Flow is used in robotics to calculate odometry [6], predict room structure [7], and depth estimation[6][7] among others. Calculation of flow is still a research topic, current techniques are limited by speed or the extent of feature displacement that they can handle.

Calculating optical flow is finding the displacement of pixels along two image. Searching all possible displacements of a pixel is extremely expensive which is why many methods were introduced which suggest ways to reduce the search space of possible disparities for a given pixel. Some methods rely on a hill-climber technique by adjusting the expected disparity through observing that of the neighboring pixels, other techniques increment predicted disparity and apply it to all pixels until the results start deteriorating. Lucas-Kanade (LK) introduced a method that relies on the spacial intensity gradient to calculate the predicted disparity. LK flow is fast to calculate but relies on the assumption that the images compared do not have high flow [8]. Speed is important for the intended application but LK results are noisy which is an important detail to account for. Optical flow is also limited to motion of features orthogonal to an intensity edges; if the robot is moving parallel to horizontal stripes LK will fail to extract information about the parallel edge motion.

As the camera moves the expected flow is predictable for objects at a certain depth, farther objects are displaced less which is why when driving one notices close objects zoom by while farther objects seem to move slower. Depth and flow are related as such on the image plane, objects on the ground plane will move in 3D space the same amount as the camera does. Those observations present a relationship between depth and disparity. Reliable flow vectors produced by LK are relatively sparse but clearly represent disparity in depth as shown in Figure 3. Flow alone is not sufficient for real depth estimation, but even with the noisiness of flow vectors, reliable depth edges can be identified with the help of Bayesian inference.

To produce the above results the image had to be downsized then convolved with a Gaussian which smooths rapid transitions and in turn improves the results of LK. One way for spreading the flow to regions with no sufficient features is through setting empty value to equal to of their nearest neighbor. This method causes issues when there are too few features. One example below shows that being the case in a campus building. To reduce noise in flow

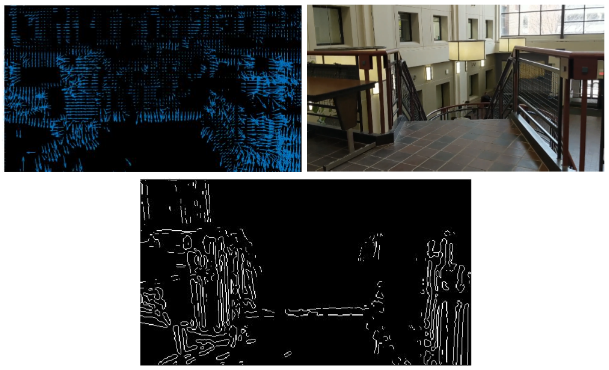

**Figure 3:** The top right images is the original image of the staircase in the East Hall study space in the Mathematics department at the University of Michigan. The top left image shows the optical flow from vectors that are extracted using Lucas-Kanade. The image at the bottom shows the edges formed by sudden change in the intensity of flow. Farther objects obviously have a lower flow. Even with less features on the ground, the edge of the staircase clearly has higher flow than objects behind it which allows for identifying the depth disparity.

magnitudes, a median filter is used to remove outliers then a Gaussian filter to build the expected relation between neighboring pixels. Using a Laplacian of Gaussian (LoG) over the resultant data provides edges where flow changes rapidly. Rapid change in flow, whether from high to low or low to high, represents a hazard that is to be avoided. The current implementation relies on hazard identification but can be expanded to differentiate between drop-off or an overhanging object.

This data alone is not sufficient for a safety application. If flow readings are too sparse the result would show a sudden change in flow between featurefull and featureless regions. To avoid this one could disregard all flow vectors with magnitudes expected of features with safe heights. This requires knowledge of motion which if available helps with calculating depth of track-able features as well.

### B. *Optical Flow & Depth*

Pixel disparity, feature depth, and camera motion are related. On Vulcan the camera is parallel to the ground plane, assuming that the lower section of the image consist in part of the ground plane one can project feature in the lower section of the image and obtain their depth using the equation below by relying on known camera parameters [6]:

$$z = \frac{H}{(2v - V)\tan(VFOV/2)}$$

Where $v$ is the vertical position of the pixel from the top of the image plane, $V$ is the height of the image plane, $VFOV$ is the vertical field of view or the angle between the focal point and the image plane edges.

One can also build a plane model. Ground plane in Vulcan's camera coordinate system would simply be represented by $n_p$:

$$n_p = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

Then depth of any pixel $p$ can be projected using the relation:

$$H = z(n_p.p)$$

where $p$ is the pixel's homogeneous coordinates in 2D. Applying motion to the points in 3D propagates them as they would be expected to propagate if they fit the model they are projected to:

$$P' = R.P + T;$$

Where $R$ and $T$ are the rotation and translation matrices representing the motion of the camera between two frames. Projecting the point back to the image frame allows for calculating the expected optical flow. In [7] a hypothesis of the room structure is updated based on observations of the flow of features. For the purpose of predicted hazards, choosing a model of an empty room and comparing the flow of features to what they would be if they fit the model would allow for determining whether the feature is one that is above, below, or approximately part of the empty room model. Observations of the difference in actual expected flow could support one the 3 hypothesis about the feature. For this method cue to be obtained motion has to be calculated.

### C. *Epipolar Constraints & Depth*

As discussed in depth in the section on CNNs, feature disparity and depth were shown to be related and obtainable in binocular images. Epipolar constraints can be also leveraged in image sequences. Similar to the section above, this would require motion be available so that the relative position of the image planes can be obtained. Although the process of such calculation is more complicated it is proven to be usable.

### D. *Odometry From Flow*

One way of obtaining odeometry is relying on the flow of features closer to the robot. Assuming the robot is on the ground plane features close to it can be tracked and projected onto the ground plane. Disparity of those 3D ground plane features can be used to obtain odometry [7]. Because the intention of this project is to avoid features at different heights it is not ideal to assume that there will always be features sufficiently close to the robot that belong to the ground plane.

Vulcan already hosts two LIDAR sensors and runs SLAM for localization. Odometry corrected by SLAM would speed up the current algorithms and allow for exploiting the cues discussed in the two previous sections. Finding a faster way to obtain transformation matrix is the only way for this application to be useful in real-time operation. The importance of having and image transform is emphasized in the section V-E.

### E. *Evaluating Hypothesis*

Overhanging objects could occupy a very small section of the image. I chose to track hypothesis of such features using a grid. This makes the hypothesis space a grid. Log odds are used as a probabilistic value in grid cells which helps bypass floating point limitations.

$$\log\left(\frac{p(h|c_1, c_2, ...)}{1 - p(h|c_1, c_2, ...)}\right) = \sum log(odds(o|c_n)$$

Where $h$ represents probability of there being a hazard in the region and $c_n$ is the $n^{th}$ cue. The use of log odds permits for separating probabilities of drop-offs and overhangs.

After obtaining a reading and hypothesizing using it, the result is reflected on the probability grid by increasing the likelihood of features that show consistent readings and decaying that of those that show no readings. Because Vulcan is moving, any new prediction does not perfectly overlap with the previous probability grid. To update the proper blocks in the grid, it has to be shifted with motion. To do so, SIFT features are extracted from every two consecutive images. The SIFT features are matched and the closest matched pairs are used to calculate the best image similarity transform. RANSAC is used to avoid the effect of falsely matched pairs, which could be matches of non-unique features that reoccur in each image.

Finding a transform is the most time consuming part of the process. The technique used for finding features is certainly not the most optimal and could be improved. If odometry was obtained from the chair it could be used to construct a rigid body transform instead. Although a rigid body transform lacks the scaling, it could be useful because of how small the disparity is between every two consecutive frames. The hypothesis calculation also accounts for minor motion and flow inaccuracies by performing Gaussian convolution on the every new prediction.

After finding the transform, it is applied to the current hypothesis space which overlaps the new and accumulated hypothesis spaces.

## VI. Results Thus Far

In methodology many different techniques were explored two of which were attempted: the use of sudden change in flow magnitudes to identify hazards, and comparison between derived flow and flow expected of a perfect room model. This section lists the steps taken towards implementing those two methods. Starting with steps needed to calculate the transformation matrix, followed by the analysis of flow, then construction of the hypothesis. This section lists what was applied and the parameters thought to best fir the application.

### A. *Feature Extraction & Matching*

Harris corner detector is used with a sigma of 1.7 and a radius of 2. Thresholding is applied to select most representative features. SIFT descriptors of the chosen corners are derived from pixels at a radius of 7 pixels around the feature. Distance between every pair of features is calculated, features with the shortest distances are chosen as matched pairs. 100 feature pairs are used to estimate similarity transform with the help of RANSAC. Figure 4 shows the chosen matches.

RANSAC runs a maximum of 1000 iterations after each of which it calculates the inliers. Any features at a max distance of 1.25 pixels from their matches are considered inliers. After a confidence level of 99% or 1000 iterations are reached the obtained transform is passed to the hypothesis calculating function which uses it to propagate the hypothesis.

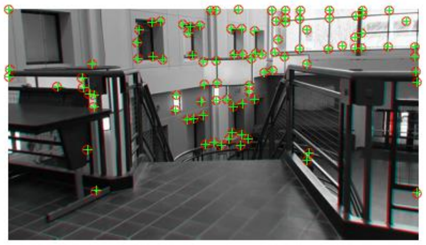

**Figure 4:** Image showing features in one image as circles and slightly shifted features in another image as plus signs. The features closest are considered matching. If zoomed in one can see those features are connected with a yellow line. Pairs are then used to calculate transform between the two images.

The initial intention was to use features to identify depth at their location. Because that requires odometry to calculate and because it is time consuming to consider all matching pairs in the image, Lucas-Kanade optical flow was instead explored. Initially time was spent on comparing flow to expected flow-described earlier as the perfect empty room comparison technique. After plenty of attempts that technique was abandoned and high flow derivative was used to detect hazard instead.

### B. Expected Flow In Empty Room

As a start the room was split into two sections the bottom of which was considered ground plane and the top part was considered a wall. Constant speed and forward direction was assumed and used to perform an initial test on the algorithms. Difficulties emerged from inaccuracies in the parameters and potentially mistakes in the algorithms. Results that were obtained are shown in Figures 5 & 6 but were not satisfactory.

After exploring ways to obtain odometry and realizing their difficulties, this method was put on-hold and focus was turned to the following method.

### C. High Flow Derivatives

Currently, as described in the methodology section, the image is convolved with a 10x10 median filter to gets rid of noise, then with a Gaussian filter of sigma 2 to smooth any outliers. Edges are then extracted using an LoG filter with a sigma of 2.

Figures 8 & 7 show results of the algorithm.

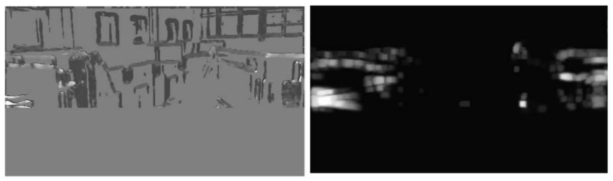

**Figure 5:** To the left the figure shows the results of analyzing two consecutive frames. White is used to indicate features that are seen to be closer than the hypothetical wall programmed and black represents those that are farther. Because we only care for overhanging objects in the top part of the image, the hypothesis only relies on the white features. The image to the right is the hypothesis accumulated from multiple iterations. The dark background is used to emphasize the certainty of the white when being displayed. A different hypothesis model was used in the final implementation of latter parts.

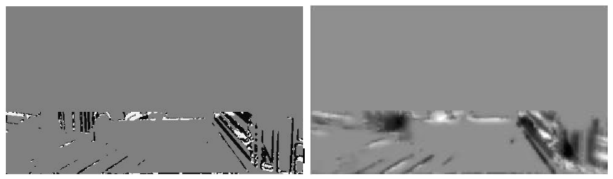

**Figure 6:** White in this image was used to represent the farther features where black represented closer ones. The right image shows results from two consecutive frames while the right one shows an accumulated hypothesis.

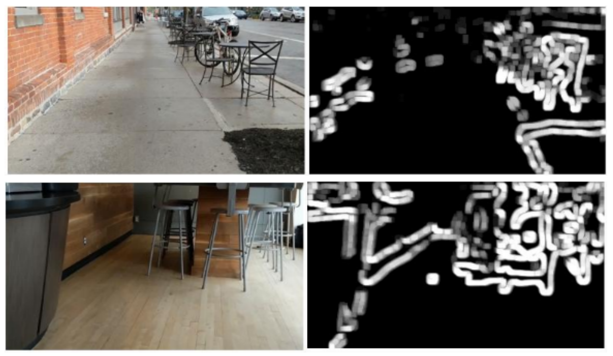

**Figure 7:** This image shows the results of the algorithm in other environments including and open street and a coffee shop. Those were significant to test that the tool does not falsely identify hazards in an open space or misidentify stools or other objects.

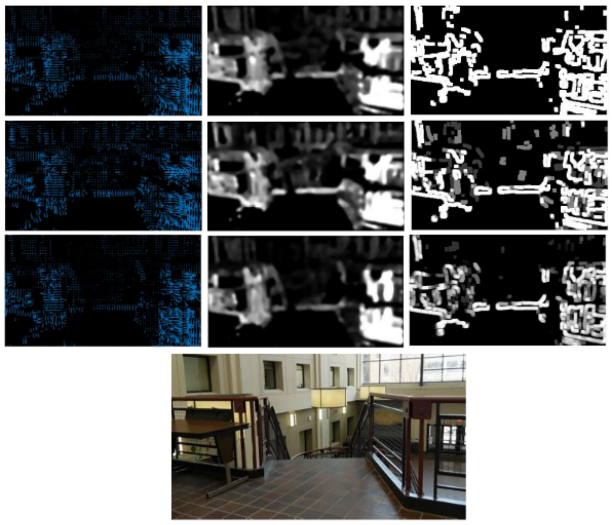

**Figure 8:** Every row, from top to bottom, in this image shows the readings from the current frame followed by the hypothesis space, with the image of the analyzed space at the bottom. The top most row shows the first frame results followed by the second the the fifth frame results. The left most image in the top three rows shows the optical flow vectors obtained, with the center image showing the refined intensities of those flows, and the last showing the hypothesis obtained after accounting to prior readings.

## VII. Discussion

The intention behind implementing the probability grid is to be able to integrate cues from several algorithms. This section will suggestion a few possible tools that could be integrated to help solidify the prediction on where hazards exist.

### A. *Expected Optical Flow*

If motion was provided to the algorithm the model of an ideal room could be used. Currently if there are few features, those features might suggest there could be sudden change in optical flow in a certain region where the change actually is a result of there being a feature island. If expected flow is available, sparse flow spots can be ignored if they show flow expected from a perfect model. If flow proved to be significant different than expected flow, the probability grid could be appended to emphasize that the region contains a hazard.

### B. *Depth From Motion*

The availability of motion also allows for solidifying prediction about the depth of features in a certain region. Using epipolar constrain algorithms one can calculate the depth of a feature. This makes it worthy to spend resources on tracking distinct features and having their depth calculation append the prediction relating to the region. Using the hypothesis probability grid would buffer the inaccuracies in localization if odometry was provided by the robot.

## References

[1] B. J. IVANECKÝ, "Depth estimation by convolutional neural networks."

[2] Y. Zhang, S. Song, E. Yumer, M. Savva, J.-Y. Lee, H. Jin, and T. Funkhouser, "Physically-based rendering for indoor scene understanding using convolutional neural networks," *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[3] C. Godard, O. Mac Aodha, and G. J. Brostow, "Unsupervised monocular depth estimation with left-right consistency," in *CVPR*, vol. 2, no. 6, 2017, p. 7.

[4] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, ISBN: 0521540518, 2004.

[5] "Optical flow." [Online]. Available: https://en.wikipedia.org/wiki/Optical_flow

[6] J. Campbell, R. Sukthankar, I. Nourbakhsh, and A. Pahwa, "A robust visual odometry and precipice detection system using consumer-grade monocular vision," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, April 2005, pp. 3421–3427.

[7] G. Tsai, C. Xu, J. Liu, and B. Kuipers, "Real-time indoor scene understanding using bayesian filtering with motion cues," in *2011 International Conference on Computer Vision*, Nov 2011, pp. 121–128.

[8] "B. lucas, t. kanade, "an iterative image registration technique with an application to stereo vision", proc. of 7th int. conf. on artifi intell., pp. 674-679, aug. 1981." [Online]. Available: https://www.ri.cmu.edu/pub_files/pub3/lucas_bruce_d_1981_1/lucas_bruce_d_1981_1.pdf